Chapter 2

Overview of Tools

2.1 What are design patterns?

Design patterns represent the best practices of experienced object-oriented software developers. Design patterns are recurring solutions to common software design problems. These solutions were discovered through trial and error by numerous software developers over a substantial period of time. Gamma, Erich, Helm, Johnson, and Vlissides (1995) catalog those best practices with 23 patterns. The authors describe each design pattern with a name and an intent, which defines what the pattern does and what design issues or problems it addresses. Just as object-oriented programming encourages code reuse, design patterns encourage design reuse. Design patterns help programmers capture the features of good programming solution and create more reusable and maintainable programs.

The following lists are the advantages and disadvantages of design patterns.

[Advantages]

- Programmers do not need to reinvent solutions to known design problems.
- Design patterns give novice developers access to the best practices proven by expert developers.
- Design patterns allow developers to think about their designs at a higher level of abstraction. For example, instead of focusing on low-level details, such as how to use inheritance, a developer can approach complex systems through a collection of design patterns that already make the best use of inheritance.
- Design patterns provide a common vocabulary for developers to discuss design. A design pattern vocabulary conveys a particular solution to a design problem more succinctly than explaining the solution with a lot of words.
- Knowledge of design patterns often brings insight in designing flexible software.

References

Fowler, Martin *Refactoring: Improving The Design of Existing Code* Reading, Mass.: Addison-Wesley, 2000.

Cooper, James W. *Java Design Patterns: A Tutorial* Reading, Mass.: Addison-Wesley, 2000.

Gamma, Erich, R. Helm, R. Johnson, and J. Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software.* Reading, Mass.: Addison-Wesley, 1995.

Kerievsky, Joshua.(2003) *DRAFT of Refactoring To Patterns.* [On-line resource]. Retrieved May 5, 2003, from the World Wide Web: < http://industriallogic.com/papers/rtp017.pdf >

Metsker, Steven J. *Design Patterns Java Workbook* Reading, Mass.: Addison-Wesley, 2002.